

WHAT IS CLAIMED IS:

1. A computer-implemented method, comprising:
receiving a request from executable code to load an
assembly, the request not including assembly version data;
5 consulting information associated with the executable
code to determine a particular version of the assembly;
and
providing the particular version of the assembly for use
by the executable code.

10

2. The method of claim 1 wherein the request corresponds
to a request to load a privatized assembly.

3. The method of claim 1 wherein the request corresponds
15 to a request to load a shared assembly.

4. The method of claim 3 wherein the shared assembly is
maintained in an assembly cache.

20 5. The method of claim 1 wherein consulting
information associated with the executable code to
determine a particular version of the assembly includes
searching for a mapping from a version independent name
provided by the executable code to a version specific
25 assembly.

6. The method of claim 5 wherein no mapping from the version independent name to a version specific assembly is present, and wherein providing the particular version of the assembly for use by the executable code comprises providing a default version.

7. The method of claim 1 wherein providing the particular version of the assembly comprises accessing a file corresponding to the assembly and loading the assembly into memory from the file.

8. The method of claim 1 wherein the information associated with the executable code includes a mapping between a version independent name provided by the executable code and a version specific file system path and filename of the particular version of the assembly, and wherein providing the particular version of the assembly comprises returning the path and filename to an assembly loading mechanism.

9. The method of claim 8 wherein the executable code is stored as an application executable file in a folder, and wherein the version of the assembly is stored as another file in the same folder.

10. The method of claim 8 wherein the filename corresponds to a file in an assembly cache.

11. The method of claim 1 wherein the information
5 associated with the executable code is derived from application manifest.

12. The method of claim 11 wherein the information
associated with the executable code is further derived from at
10 least one assembly manifest..

13. The method of claim 1 wherein the information
associated with the executable code is constructed during a
pre-execution initialization phase.

14. The method of claim 1 wherein the information
associated with the executable code is persisted into a non-
volatile memory.

15. A computer-readable medium having computer-executable
20 instructions for performing the method of claim 1.

16. A computer-implemented method, comprising:
interpreting dependency information associated with
executable code, the dependency information identifying at
least one particular version of an assembly; and

5 associating with the executable code at least one mapping
based on the dependency information, each mapping relating a
version independent assembly name that the executable code may
provide to a version specific assembly identified in the
dependency information.

10 17. The method of claim 16 wherein the dependency
information is provided in an application manifest associated
with the executable code.

15 18. The method of claim 17 wherein the application
manifest is associated with the executable code by being stored
in a common folder with an application executable file that
corresponds to the executable code.

20 19. The method of claim 16 wherein at least one
mapping maps a version independent name to an assembly
stored in a common folder with an application executable
file that corresponds to the executable code.

20. The method of claim 16 wherein at least one mapping maps a version independent name to a shared assembly in an assembly cache.

5 21. The method of claim 16 wherein the dependency information provided by the executable code corresponds to an assembly having an assembly manifest associated therewith, and further comprising, interpreting the assembly manifest.

10 22. The method of claim 21 wherein the assembly manifest specifies that a particular version of an assembly be replaced with another version of that assembly.

15 23. The method of claim 21 wherein the assembly manifest specifies at least one particular version of another assembly on which the assembly having an assembly manifest is dependent.

20 24. The method of claim 16 wherein the dependency information is interpreted in response to receiving a request to execute the executable code.

25 25. The method of claim 16 wherein the at least one mapping is maintained in an activation context, and further comprising, persisting the activation context.

26. The method of claim 25 wherein associating with the executable code the at least one mapping comprises retrieving a persisted activation context.

5

27. The method of claim 25 wherein associating with the executable code the at least one mapping comprises constructing a new activation context.

10 28. The method of claim 27 wherein the new activation context is constructed upon determining that an activation context does not exist.

15 29. The method of claim 27 wherein the new activation context is constructed upon determining that an existing activation may not be not coherent with current policy.

20 30. The method of claim 16 further comprising, running the executable code, receiving a request from the executable code to load an assembly, the request including data corresponding to a version independent name of the assembly and providing a particular version of the assembly for use by the executable code based on a mapping therefor.

25

31. A computer-readable medium having computer-executable instructions for performing the method of claim 16.

32. A computer-readable medium having stored thereon a
5 data structure, comprising:

a first set of data comprising a name of an assembly;

a second set of data comprising a version of the assembly;

a third set of data comprising at least one item of the
assembly; and

10 a fourth set of data comprising binding path data to each
item in the third set of data.

33. The data structure of claim 32, wherein the binding
path data comprises a location of a dynamic link library.

15 34. The data structure of claim 32, wherein the binding
path data comprises an object class identifier.

35. The data structure of claim 32, wherein the binding
20 path data comprises a programmatic identifier.

36. The data structure of claim 32 further comprising, a
fifth set of data comprising data corresponding to at least one
dependency on an assembly.

25

37. The data structure of claim 32 further comprising, a fifth set of data comprising data corresponding to a Windows® class.

5 38. The data structure of claim 32 further comprising, a fifth set of data comprising data corresponding to a global name.

10 39. A computer-readable medium having stored thereon a data structure, comprising:

 a first set of data comprising a version independent name of an assembly; and

 a second set of data comprising a filename path to a specific version of the assembly;

15 wherein the second set of data is associated with the first set of data such that a reference to the version independent name in the first set of data is mapped to the specific version of the assembly via the second set of data.

40. The data structure of claim 39 further comprising, a third set of data comprising a version independent object class name, a fourth set of data comprising an assembly name corresponding to a file that contains an object class that
5 corresponds to the object class name in the third set of data, and a fifth set of data comprising a version specific name that corresponds to the third set of data.

41. A computer-readable medium having stored thereon a
10 data structure, comprising:

a first set of data comprising a version independent object class name;

a second set of data comprising an assembly name corresponding to a file that contains an object class that
15 corresponds to the object class name in the first set of data; and

a third set of data comprising a version specific name that corresponds to the first set of data such that a reference to the version independent name in the first set of data is
20 mapped to the specific version of the object class.

42. A system in a computing environment, comprising:
an initialization mechanism configured to interpret
dependency data associated with executable code, the
dependency data corresponding to at least one assembly
5 version on which the executable code depends;

an activation context, the activation context
associated with the executable code and constructed by the
initialization mechanism based on the dependency data, the
activation context relating at least one version
10 independent assembly identifier to a version specific
assembly; and

a version matching mechanism configured to access the
activation context to relate a version independent request
from the executable code to a version specific assembly.

15

43. The system of claim 42 wherein the dependency
data is included in executable code manifest.

44. The system of claim 42 wherein the dependency
20 data is included in an XML file.

45. The system of claim 42 wherein the
initialization mechanism persists the activation context.

46. The system of claim 42 further comprising, an assembly loading mechanism configured to communicate with the executable code and the version matching mechanism to load the version specific assembly upon a request by the executable code to load a requested assembly, wherein the request does not include version specific data.

47. The system of claim 46 wherein the assembly loading mechanism loads the version specific assembly from an assembly cache.

48. The system of claim 42 wherein the dependency data identifies an assembly that has assembly dependency data associated therewith, the assembly dependency data corresponding to at least one other assembly version on which the assembly depends, and wherein the initialization mechanism adds information that corresponds to the assembly dependency data to the activation context.

49. A computer-readable medium having computer-executable modules configured to implement the system of claim 42.